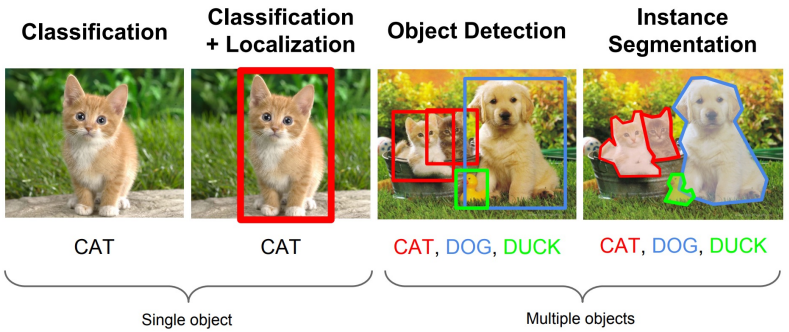
AA 2022/2023

Machine Learning for Modelling: *Supervised Learning*

Simone Bianco

1

CV tasks



2

A road map of object detection

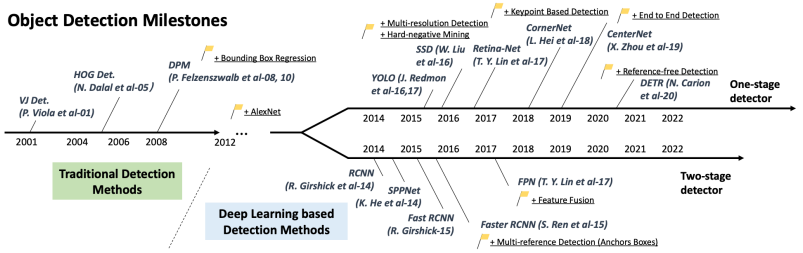


Fig. 2: A road map of object detection. Milestone detectors in this figure: VJ Det. [10, 11], HOG Det. [12], DPM [13–15], RCNN [16], SPPNet [17], Fast RCNN [18], Faster RCNN [19], YOLO [20–22], SSD [23], FPN [24], Retina-Net [25], CornerNet [26], CenterNet [27], DETR [28].

3

A road map of object detection

Object detection has generally gone through two historical periods:

- Traditional object detection period (before 2014)
- Deep learning-based detection period (after 2014)

In the deep learning era, object detection can be grouped into two genres:

- Two-stage detection: frames the detection as a “coarse-to-fine” process
- One-stage detection: frames the detection as “complete in one step”

4

A road map of object detection

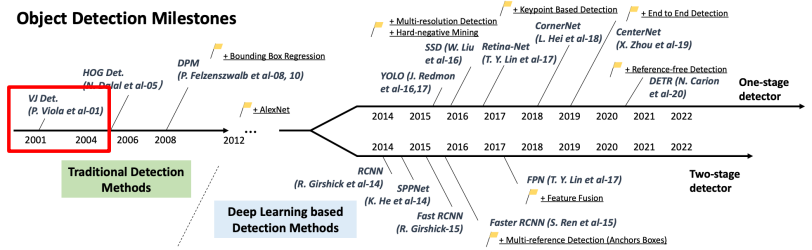


Fig. 2: A road map of object detection. Milestone detectors in this figure: VJ Det. [10, 11], HOG Det. [12], DPM [13–15], RCNN [16], SPPNet [17], Fast RCNN [18], Faster RCNN [19], YOLO [20–22], SSD [23], FPN [24], Retina-Net [25], CornerNet [26], CenterNet [27], DETR [28].

5

A road map of object detection

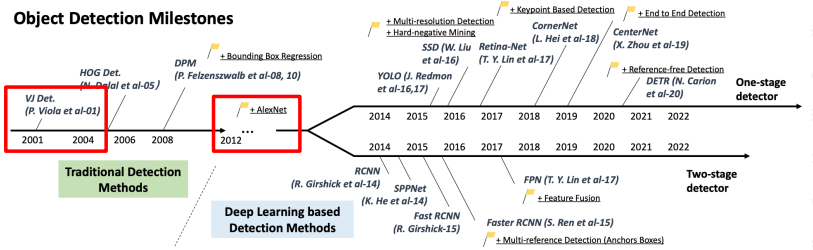
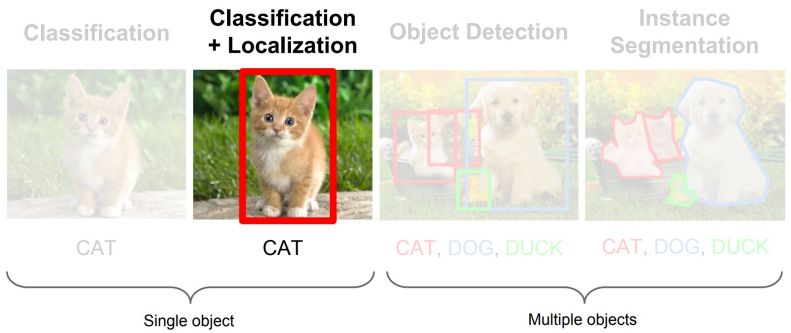


Fig. 2: A road map of object detection. Milestone detectors in this figure: VJ Det. [10, 11], HOG Det. [12], DPM [13–15], RCNN [16], SPPNet [17], Fast RCNN [18], Faster RCNN [19], YOLO [20–22], SSD [23], FPN [24], Retina-Net [25], CornerNet [26], CenterNet [27], DETR [28].

6

CV tasks



7

Task: classification + localization

Classification: C classes
Input: Image
Output: Class label
Evaluation metric: Accuracy

→ CAT

Localization:
Input: Image
Output: Box in the image (x, y, w, h)
Evaluation metric: Intersection over Union

→ (x, y, w, h)

Classification + Localization: Do both

8

Task: classification + localization

Input: image

Neural Net

Output:
Box coordinates
(4 numbers)

Correct output:
box coordinates
(4 numbers)

Loss:
L2 distance

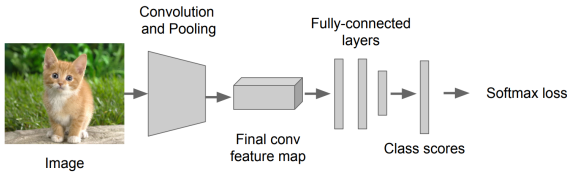
Only one object,
simpler than detection

9

Task: classification + localization

A simple recipe (a.k.a. Localization as regression):

- step 1: Train a classification model (e.g., AlexNet, VGG, GoogleNet)

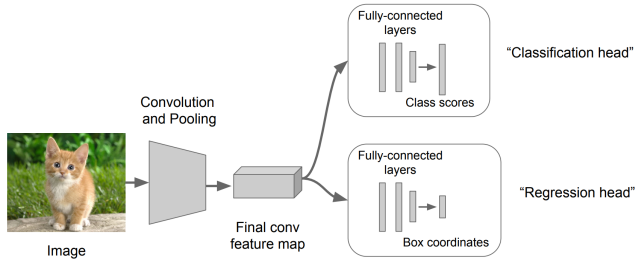


10

Task: classification + localization

A simple recipe (a.k.a. Localization as regression):

- step 1: Train a classification model (e.g., AlexNet, VGG, GoogleNet)
- step 2: Attach a new fully-connected "regression head" to the network

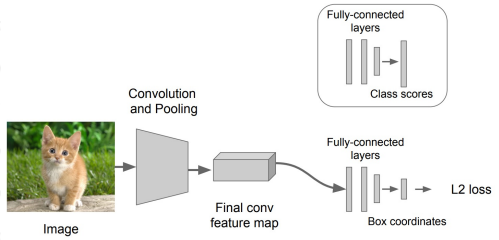


11

Task: classification + localization

A simple recipe (a.k.a. Localization as regression):

- step 1: Train a classification model (e.g., AlexNet, VGG, GoogleNet)
- step 2: Attach a new fully-connected "regression head" to the network
- Step 3: Train the regression head only with SGD and L2 loss

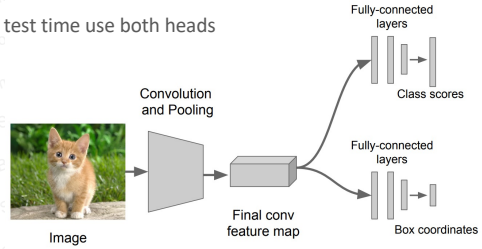


12

Task: classification + localization

A simple recipe (a.k.a. Localization as regression):

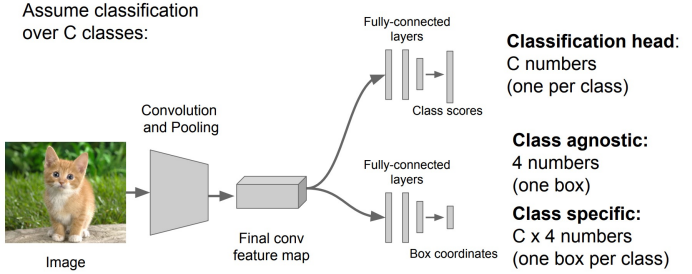
- step 1: Train a classification model (e.g., AlexNet, VGG, GoogleNet)
- step 2: Attach a new fully-connected "regression head" to the network
- Step 3: Train the regression head only with SGD and L2 loss
- Step 4: At test time use both heads



13

Per-class vs class agnostic regression

Assume classification over C classes:



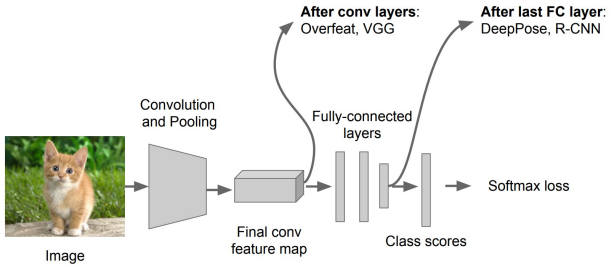
Classification head:
C numbers
(one per class)

Class agnostic:
4 numbers
(one box)

Class specific:
C x 4 numbers
(one box per class)

14

Where to attach the regression head?



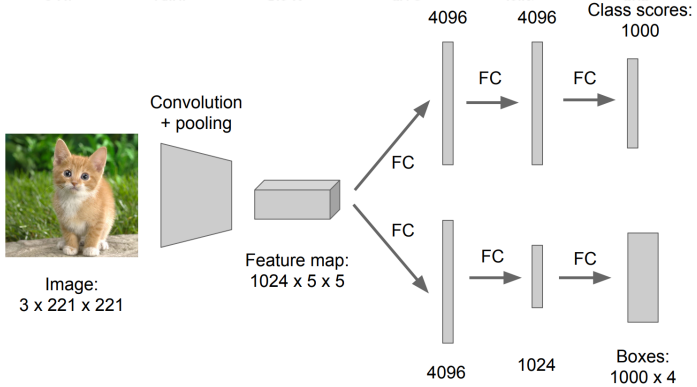
15

(Efficient) sliding window

- Run classification + regression network at multiple locations on a high-resolution image
- Convert fully-connected layers into convolutional layers for efficient computation
- Combine classifier and regressor predictions across all scales for final prediction

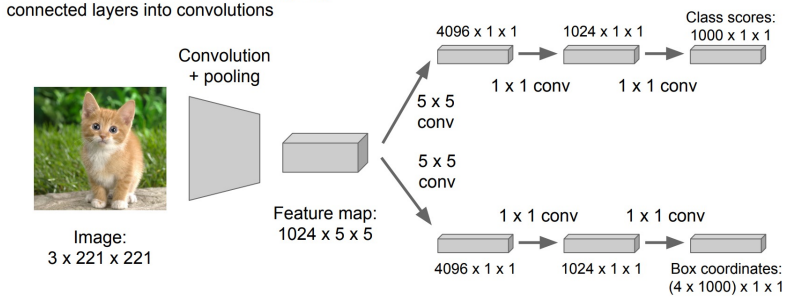
16

(Efficient) sliding window



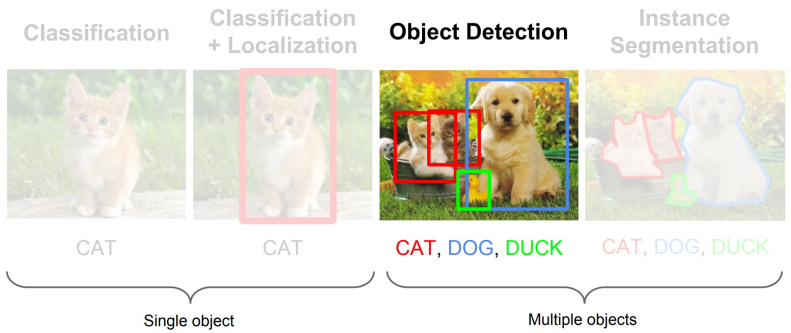
17

(Efficient) sliding window



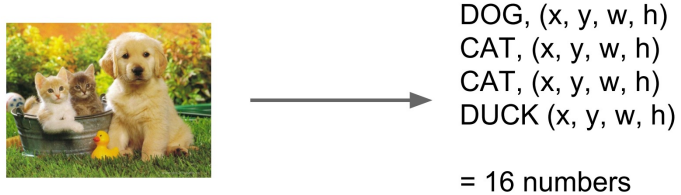
18

CV tasks



19

Detection as regression?



20

Detection as regression?

DOG, (x, y, w, h)
CAT, (x, y, w, h)

= 8 numbers

21

Detection as regression?

CAT, (x, y, w, h)
CAT, (x, y, w, h)
….
CAT (x, y, w, h)

= many numbers

We would need variable sized outputs…

22

Detection as classification

CAT? NO

DOG? NO

23

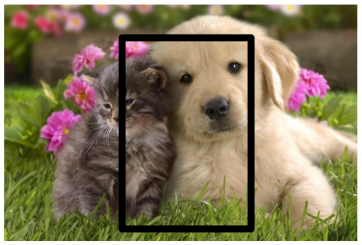
Detection as classification

CAT? YES!

DOG? NO

24

Detection as classification



CAT? NO

DOG? NO

25

Detection as classification

- **Problem**: Need to test many positions and scales
- **Solution**: If your classifier is fast enough, just do it

26

Detection as classification

- **Problem**: Need to test many positions and scales
- **Solution**: If your classifier is fast enough, just do it

More likely:

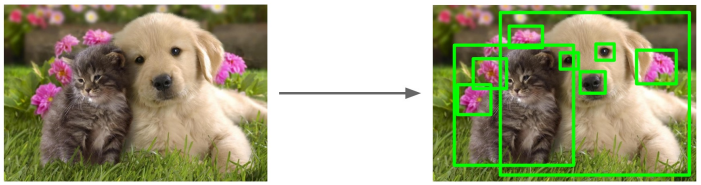
- **Problem**: Need to test many positions and scales, and need to use a computationally demanding classifier (e.g., CNN)
- **Solution**: Only look at a tiny subset of possible positions

27

Detection as classification

Region proposals:

- Find "blobby" image regions as the are likely to contain objects
- "Class-agnostic" object detector
- Look for "blob-like" regions



28

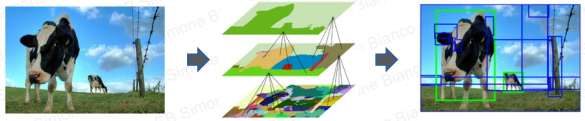
Region proposals (object proposals)

29

Selective search

1. Initial image regions
2. Similarity computation
3. Greedy bottom-up grouping

NOT NEURAL

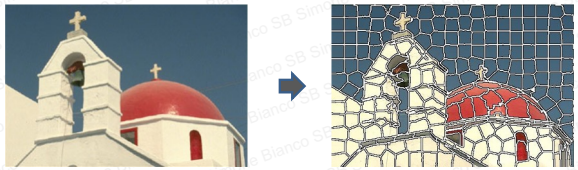


[1] Uijlings, J. R., van de Sande, K. E., Gevers, T., & Smeulders, A. W. (2013). Selective search for object recognition. *International journal of computer vision*,*104*(2), 154-171.

30

Selective search: step 1. Initial image regions

- Graph-based segmentation of the input image
  - Felzenszwalb, P. F. and Huttenlocher, D. P. (2004). Efficient graph-based image segmentation. International Journal of Computer Vision, 59(2):167–181.
- Exploiting the concept of **superpixels**



31

Selective search: step 2. Similarity computation

- Similarity between all pairs of neighboring regions
- Criteria:
  - **Color**: A color histogram for each color channel is computed.
  - **Texture**: Gaussian derivatives are computed into orientation histograms.
  - **Size**: Smaller regions are given a larger incentive for merging.
  - **Fill**: A pair of regions where one encloses the other, is given higher merging likelihood.

32

Selective search: step 3. Greedy bottom-up grouping

- The two most similar regions are merged
- Similarity with the other regions is updated
- The regions with lower rank are removed from the list of generated object proposals.

- Objective: produce multiple overlapping proposals
  - A **wheel** is an object by itself
  - It is also part of a bigger object «**car**»

33

Selective search: see it in action

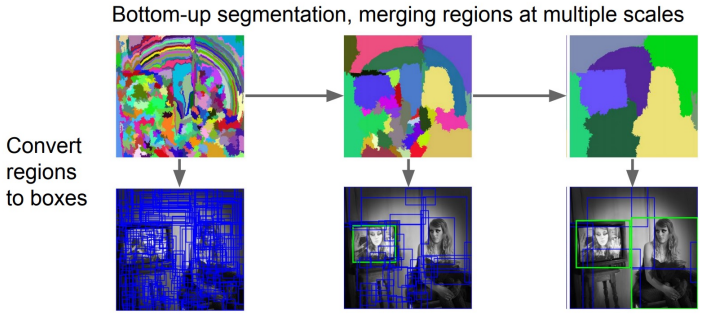
34

Selective search: see it in action

Results in approx. 2k proposals per image

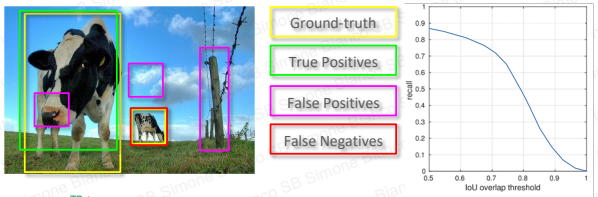
35

Selective search: see it in action



36

Evaluating object proposals



- **Recall** only: $^{TP}/_{(TP+FN)}$
  - True Positives:
    Counted when the proposal overlaps with the ground truth for more than a predefined threshold
  - False Positives:
    Ignored: that's a problem for the classifier that comes next
- Overlap between 60% and 80% proved to be predictive of detector accuracy, for DPM [1].

[1] Hosang, J., Benenson, R., Dollár, P., & Schiele, B. (2015). What makes for effective detection proposals?. arXiv preprint arXiv:1502.05082.
[2] Hosang, J., Benenson, R., & Schiele, B. (2014). How good are detection proposals, really?. arXiv preprint arXiv:1406.6962.

37

Comparison of object proposals

Method	Approach	Outputs Segments	Outputs Score	Control #proposals	Time (sec.)	Repea-tability	Recall Results	Detection Results
Bing [18]	Window scoring		✓	✓	0.2	⋆⋆⋆	⋆	·
CPMC [19]	Grouping	✓	✓	✓	250	-	⋆⋆	⋆
EdgeBoxes [20]	Window scoring		✓	✓	0.3	⋆⋆	⋆⋆⋆	⋆⋆⋆
Endres [21]	Grouping	✓	✓	✓	100	-	⋆⋆⋆	⋆⋆
Geodesic [22]	Grouping	✓		✓	1	⋆	⋆⋆⋆	⋆⋆
MCG [23]	Grouping	✓	✓	✓	30	⋆	⋆⋆⋆	⋆⋆⋆
Objectness [24]	Window scoring		✓	✓	3	·	⋆	·
Rahtu [25]	Window scoring		✓	✓	3	·	·	⋆
RandomizedPrim's [26]	Grouping	✓		✓	1	⋆	⋆	⋆⋆
Rantalankila [27]	Grouping	✓		✓	10	⋆⋆	·	⋆⋆
Rigor [28]	Grouping	✓		✓	10	⋆	⋆⋆	⋆⋆
SelectiveSearch [29]	Grouping	✓	✓	✓	10	⋆⋆	⋆⋆⋆	⋆⋆⋆
Gaussian				✓	0	·	·	⋆
SlidingWindow				✓	0	⋆⋆⋆	·	·
Superpixels		✓			1	⋆	·	·
Uniform				✓	0	·	·	·

Table 1: Comparison of different detection proposal methods. Grey check-marks indicate that the number of proposals is controlled by indirectly adjusting parameters. Repeatability, quality, and detection rankings are provided as rough summary of the experimental results: "-" indicates no data, "·", "⋆", "⋆⋆", "⋆ ⋆ ⋆" indicate progressively better results. These guidelines were obtained based on experiments presented in sections §3, §4, and §5, respectively.

Summary and all citations from [1]:
[1] Hosang, J., Benenson, R., Dollár, P., & Schiele, B. (2015). What makes for effective detection proposals?. arXiv preprint arXiv:1502.05082.

38

AA 2022/2023

Region-based convolutional neural networks (R-CNN)

39

R-CNN (Regions with CNN features)

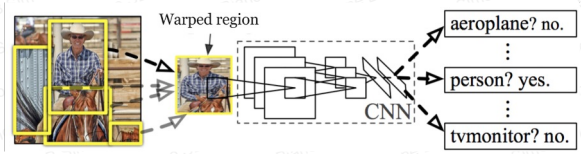
- Semi-neural approach for object detection:
  - Object proposal from Selective Search (non-neural)
  - Classification of each candidate using neural features

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).

40

R-CNN: Classification step

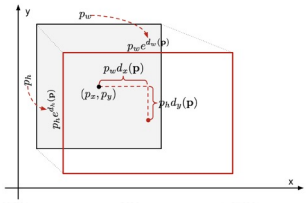
- Each candidate is warped to a common size
- Neural features are extracted from each candidate
  - Alexnet pretrained on IMAGENET (1000 classes)
  - Alexnet fine-tuned on the domain of interest
- Support Vector Machine (SVM) for final classification
  - Uses neural features as input
  - Binary classification for each possible class (+ unknown class)



41

R-CNN: Extra step 1: Bounding box regression

- Estimate a correction for the candidate location:
  - Scale-invariant transformation for the bounding box center
  - Log-scale transformation for the bounding box sizes (height and width)



Input: candidate coordinates

$$\mathbf{p} = (p_x, p_y, p_w, p_h)$$

Output: corrected coordinates

$$\mathbf{g} = (g_x, g_y, g_w, g_h)$$

Target:

$$\begin{aligned} t_x &= (g_x - p_x)/p_w \\ t_y &= (g_y - p_y)/p_h \\ t_w &= \log(g_w/p_w) \\ t_h &= \log(g_h/p_h) \end{aligned}$$

42

42

R-CNN: Extra step 2: Non-maximum suppression

- Objective: remove multiple bounding boxes for the same object

1. Sort all the bounding boxes by confidence score
2. Discard boxes with low confidence scores
3. Greedily select the one with the highest score
4. Discard boxes with high overlap with previously selected one

Before non-max suppression

After non-max suppression

43

R-CNN: Extra step 3: Hard negative mining

- Automatically select useful negative samples
  - e.g., if I want to find a **face**:
    - A plain region in the sky is an **easy** negative (useless)
    - A face-looking pattern in foliage is a **hard** negative

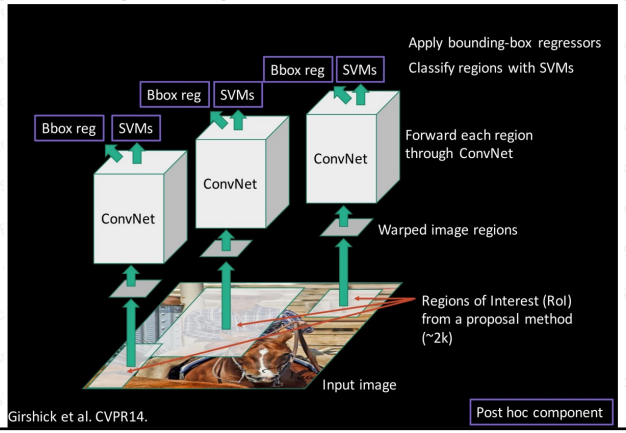
1. Train the model with **random** negatives
2. Run the model on an annotated validation set
3. Select all false positives as (hard) negative examples

- Be careful!

  The validation set must be exhaustively annotated!

44

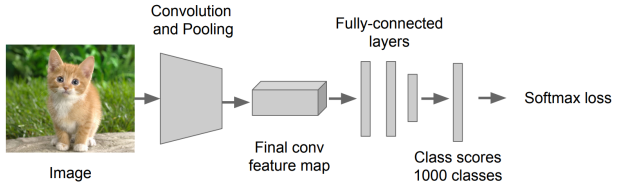
R-CNN: putting all together



45

R-CNN training

Step 1: Train a classification model for ImageNet (AlexNet)

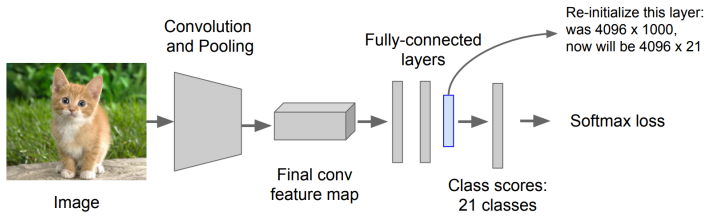


46

R-CNN training

Step 1: Train a classification model for ImageNet (AlexNet)

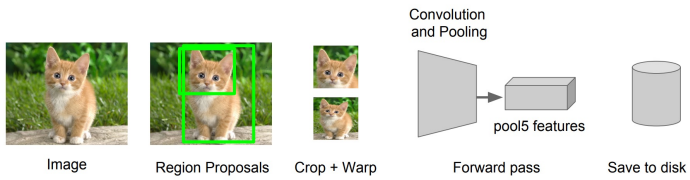
Step 2: Fine-tune model for detection:
- Instead of 1000 ImageNet classes, want 20 object classes + background
- Throw away final fully-connected layer, reinitialize from scratch
- Keep training model using positive/negative regions from detection images



47

R-CNN training

Step 3: Extract features:
- Extract region proposals for all images
- For each region: warp to CNN input size, run forward through CNN, save pool5 features to disk

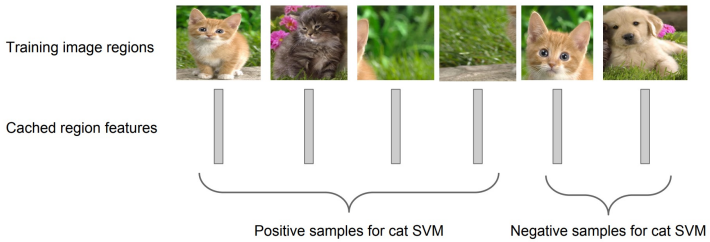


48

R-CNN training

Step 3: Extract features:
- Extract region proposals for all images
- For each region: warp to CNN input size, run forward through CNN, save pool5 features to disk

Step 4: Train one binary SVM per class (OVA) to classify region features

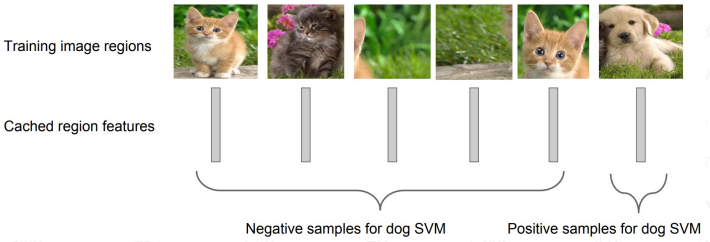


49

R-CNN training

Step 3: Extract features:
- Extract region proposals for all images
- For each region: warp to CNN input size, run forward through CNN, save pool5 features to disk

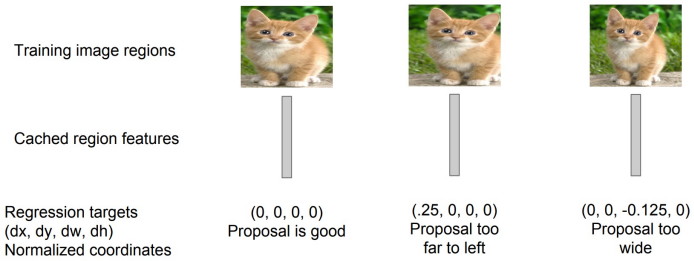
Step 4: Train one binary SVM per class (OVA) to classify region features



50

R-CNN training

Step 5 (bbox regression): For each class, train a linear regression model to map from cached features to offsets to GT boxes to coper for "slightly wrong" proposals



51

Object detection: Datasets

	PASCAL VOC (2010)	ImageNet Detection (ILSVRC 2014)	MS-COCO (2014)
Number of classes	20	**200**	80
Number of images (train + val)	~20k	**~470k**	~120k
Mean objects per image	2.4	1.1	**7.2**

52

Object detection: Datasets

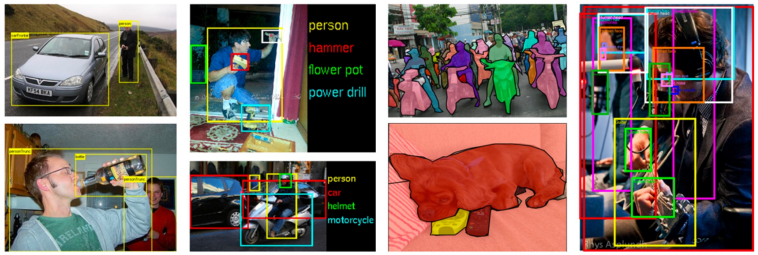


Fig. 4. Some example images and annotations in (a) PASCAL-VOC07, (b) ILSVRC, (c) MS-COCO, and (d) Open Images.

53

Object detection: Datasets

Dataset	train		validation		trainval		test	
	images	objects	images	objects	images	objects	images	objects
VOC-2007	2,501	6,301	2,510	6,307	5,011	12,608	4,952	14,976
VOC-2012	5,717	13,609	5,823	13,841	11,540	27,450	10,991	-
ILSVRC-2014	456,567	478,807	20,121	55,502	476,688	534,309	40,152	-
ILSVRC-2017	456,567	478,807	20,121	55,502	476,688	534,309	65,500	-
MS-COCO-2015	82,783	604,907	40,504	291,875	123,287	896,782	81,434	-
MS-COCO-2018	118,287	860,001	5,000	36,781	123,287	896,782	40,670	-
OID-2018	1,743,042	14,610,229	41,620	204,621	1,784,662	14,814,850	125,436	625,282

TABLE 1
Some well-known object detection datasets and their statistics.

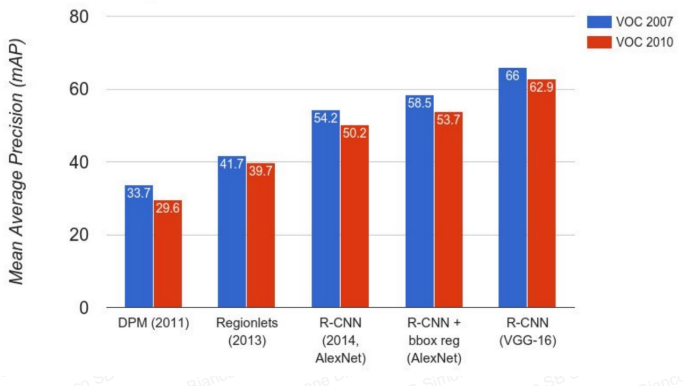
54

Object detection: Evaluation

- We use a metric called “mean average precision” (mAP)
- Compute average precision (AP) separately for each class, then average over classes
  (Combine all detections from all test images to draw a precision/curve for each class; AP is area under the curve)
- A detection is a true positive if it has a IoU with a ground-truth box greater than some threshold (usually 0.5) → we talk about mAP @ 0.5
- mAP is a number from 0 to 100, where higher is better

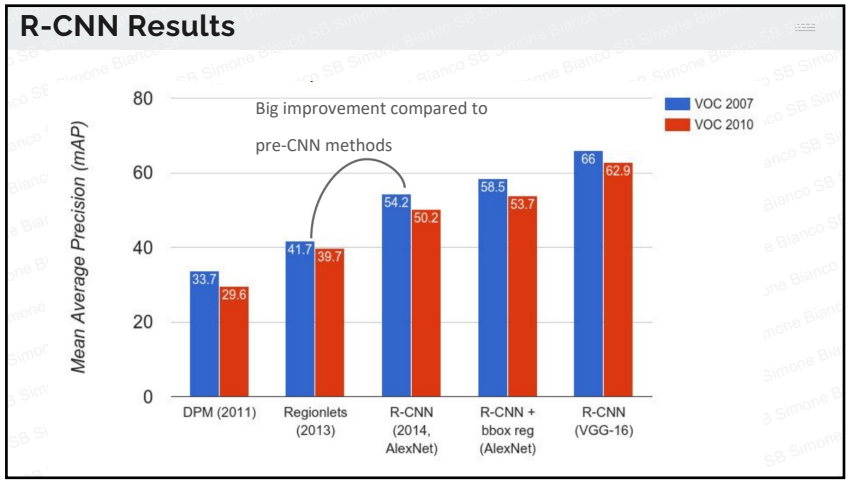
55

R-CNN Results



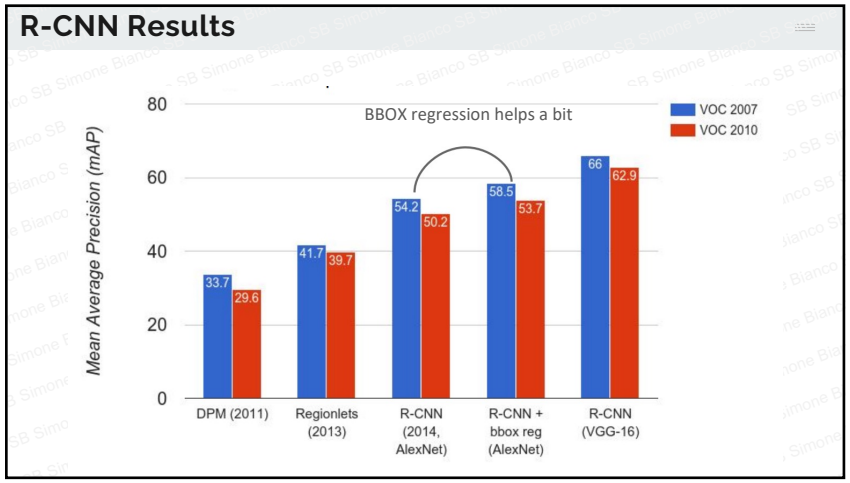
56

R-CNN Results



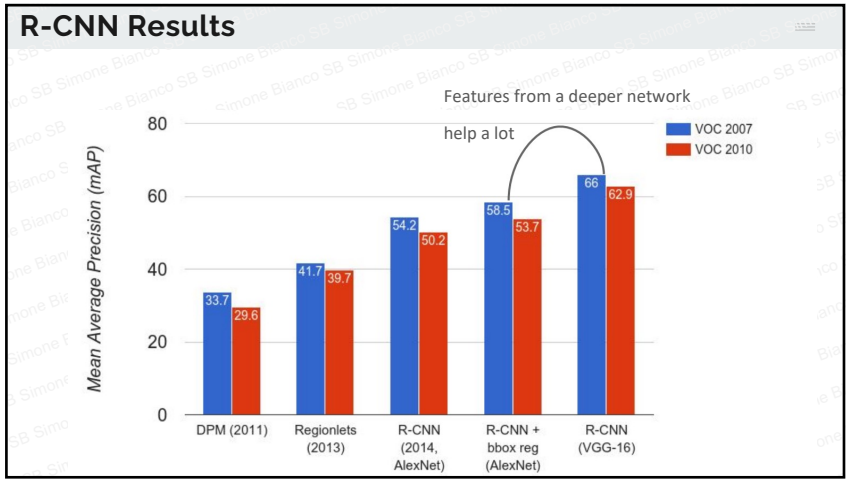
57

R-CNN Results



58

R-CNN Results



59

R-CNN Bottlenecks

1. Selective search is slow
   - Ten seconds per image
2. Feature extraction is inefficient
   - 2000 candidate windows per image
   - Extracting neural features for each candidate
   - Mostly overlapped, so the same regions will be processed multiple times

60

R-CNN Problems

1. SVMs and regressors are post-hoc: CNN features are not updated in response to SVMs and regressors
2. Complex multistage training pipeline

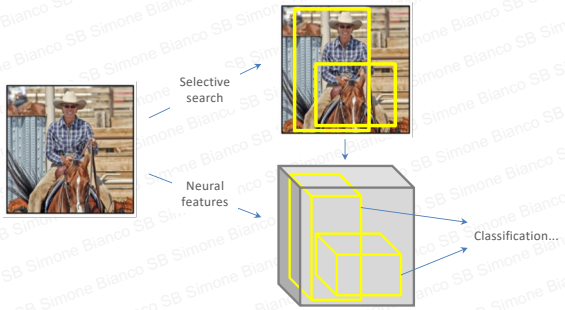
61

Fast R-CNN

62

Fast R-CNN

- Compute dense features once, and sample them based on proposals

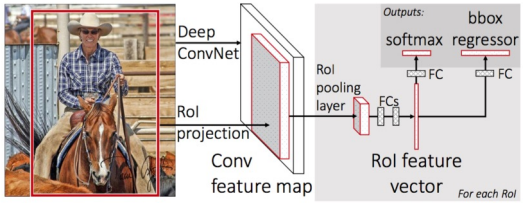


Girshick, R. (2015). Fast r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 1440-1448).

63

Fast R-CNN

- Compute dense features once, and sample them based on proposals
- Bring sampled feature blocks to a common size
  - RoI pooling layer (Region of Interest)



Girshick, R. (2015). Fast r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 1440-1448).

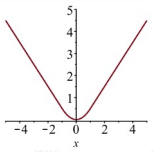
64

Fast R-CNN Classification

- Replace SVM with softmax (neural) classification
- Optimize the sum of two losses:
  - Classification $\mathcal{L}_{\text{cls}}(p, u) = -\log p_u$
  - BBox correction $\mathcal{L}_{\text{box}}(t^u, v) = \sum_{i \in \{x,y,w,h\}} L_1^{\text{smooth}}(t_i^u - v_i)$

p: prediction confidence
u: ground truth class

t: corrected bounding box
v: true bounding box



$y = L_1^{\text{smooth}}(x)$

65

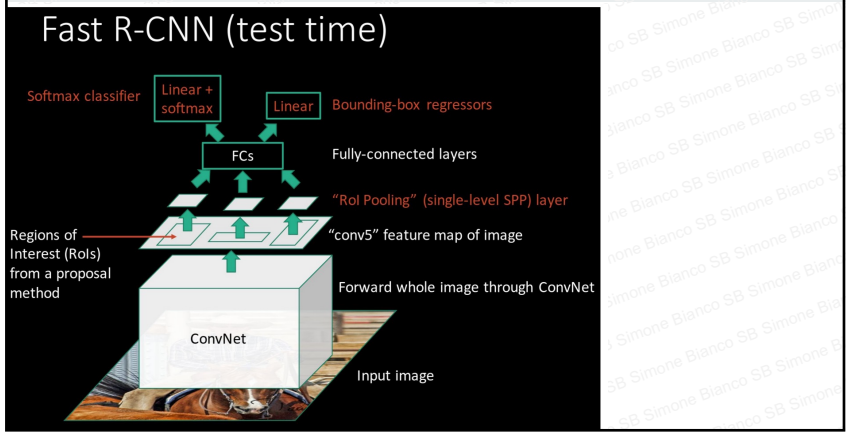
Fast R-CNN Bottleneck

1. Selective search is slow
   - Ten seconds per image
2. Feature extraction is inefficient
   - 2000 candidate windows per image
   - Extracting neural features for each candidate
   - Mostly overlapped, so the same regions will be processed multiple times

SOLVED

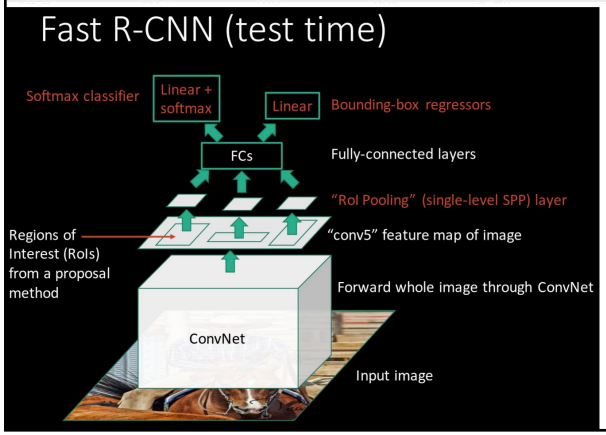
66

Fast R-CNN



67

Fast R-CNN

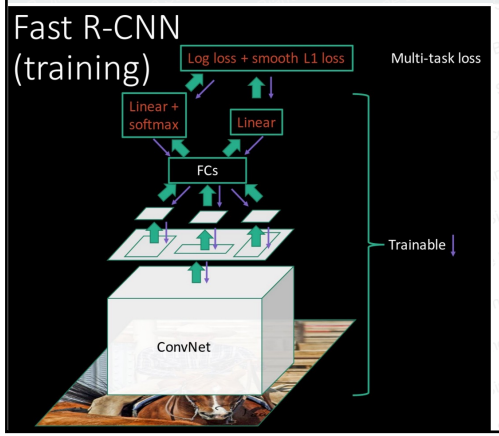


R-CNN problem #1: Slow at test-time due to independent forward passes of the CNN

Solution: Share computation of convolutional layers between proposals for an image

68

Fast R-CNN



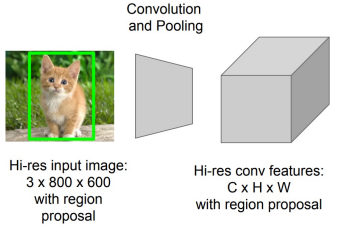
R-CNN Problem #2: Post-hoc training: CNN not updated in response to final classifiers and regressors

R-CNN Problem #3: Complex training pipeline

Solution: Just train the whole system end-to-end all at once!

69

Fast R-CNN: RoI pooling

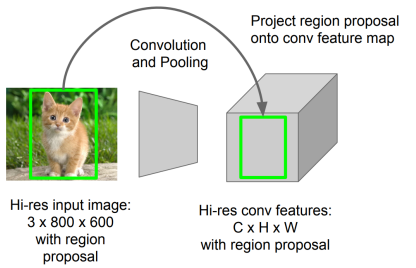


Fully-connected layers

Problem: Fully-connected layers expect low-res conv features: C x h x w

70

Fast R-CNN: RoI pooling

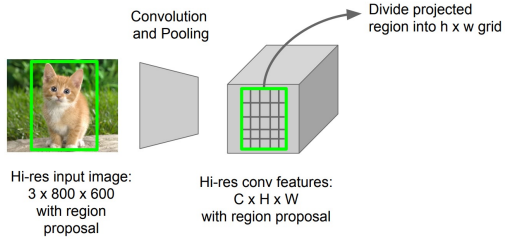


Fully-connected layers

Problem: Fully-connected layers expect low-res conv features: C x h x w

71

Fast R-CNN: RoI pooling

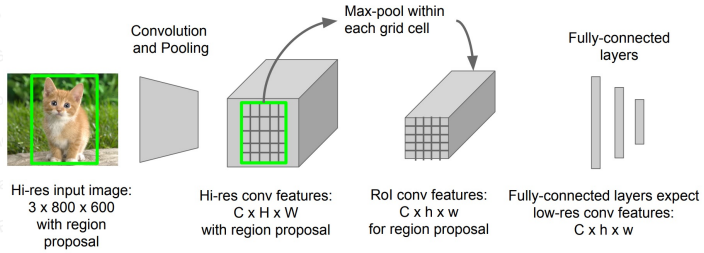


Fully-connected layers

Problem: Fully-connected layers expect low-res conv features: C x h x w

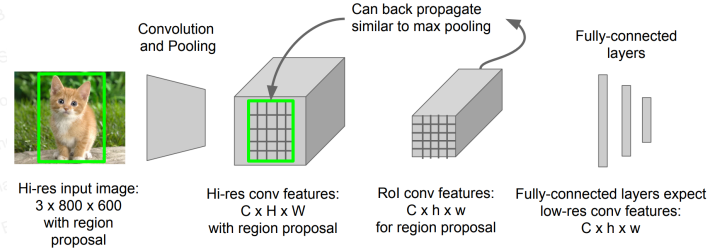
72

Fast R-CNN: RoI pooling



73

Fast R-CNN: RoI pooling



74

Fast R-CNN: Results

Faster!

	R-CNN	Fast R-CNN
Training Time:	84 hours	**9.5 hours**
(Speedup)	1x	**8.8x**

75

Fast R-CNN: Results

Faster!

FASTER!

	R-CNN	Fast R-CNN
Training Time:	84 hours	**9.5 hours**
(Speedup)	1x	**8.8x**
Test time per image	47 seconds	**0.32 seconds**
(Speedup)	1x	**146x**

76

Fast R-CNN: Results

		R-CNN	Fast R-CNN
Faster!	Training Time:	84 hours	**9.5 hours**
	(Speedup)	1x	**8.8x**
FASTER!	Test time per image	47 seconds	**0.32 seconds**
	(Speedup)	1x	**146x**
Better!	mAP (VOC 2007)	66.0	**66.9**

77

Fast R-CNN: Problem

	R-CNN	Fast R-CNN
Test time per image	47 seconds	**0.32 seconds**
(Speedup)	1x	**146x**
Test time per image with Selective Search	50 seconds	**2 seconds**
(Speedup)	1x	**25x**

Test-time speeds don't include region proposals

78

Fast R-CNN: Solution

	R-CNN	Fast R-CNN
Test time per image	47 seconds	**0.32 seconds**
(Speedup)	1x	**146x**
Test time per image with Selective Search	50 seconds	**2 seconds**
(Speedup)	1x	**25x**

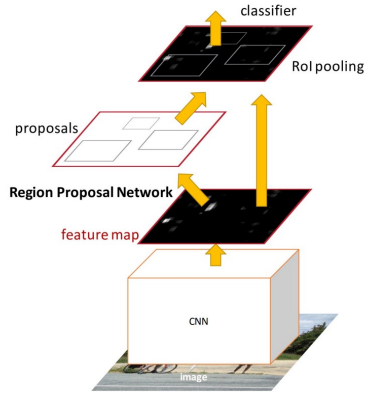
Test-time speeds don't include region proposals
→ Just make the CNN do region proposals too!

79

Faster R-CNN

80

Faster R-CNN



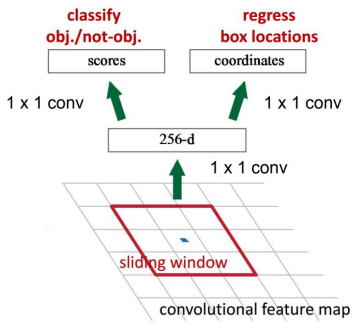
Insert a **Region Proposal Network (RPN)** after the last convolutional layer

RPN trained to produce region proposals directly; no need for external region proposals!

After RPN, use RoI Pooling and an upstream classifier and bbox regressor just like Fast R-CNN

81

Faster R-CNN: RPN



Slide a small window on the feature map

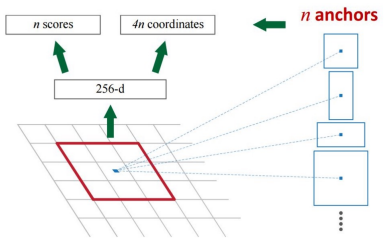
Build a small network for:
- classifying object or not-object, and
- regressing bbox locations

Position of the sliding window provides localization information with reference to the image

Box regression provides finer localization information with reference to this sliding window

82

Faster R-CNN: RPN



Use **N anchor boxes** at each location

Anchors are **translation invariant**: use the same ones at every location

Regression gives offsets from anchor boxes

Classification gives the probability that each (regressed) anchor shows an object

83

Faster R-CNN

- Replace Selective Search with Region Proposal Network (RPN)
  - Direct regression of bounding boxes from the input image
- Can be trained in two iterating phases, or end-to-end

Loss:

$$\frac{1}{N_{\text{cls}}} \sum_i \mathcal{L}_{\text{cls}}(p_i, p_i^*) + \frac{\lambda}{N_{\text{box}}} \sum_i p_i^* \cdot L_1^{\text{smooth}}(t_i - t_i^*)$$

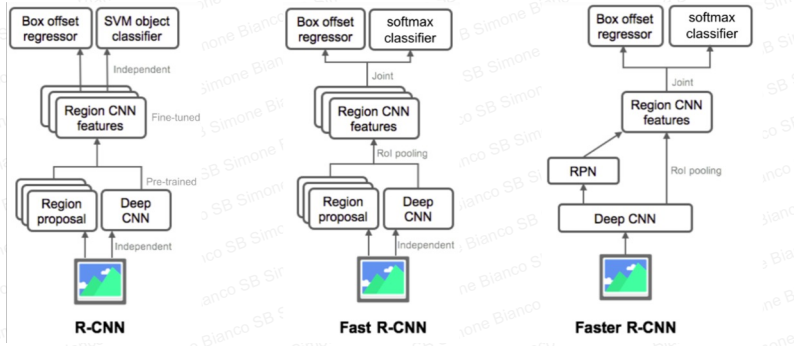
p: class prediction confidence
p*: class ground truth (binary)

t: bbox prediction
t*: true bounding box

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems (pp. 91-99).

84

Summary of models in the R-CNN family



85

Faster R-CNN: Results

	R-CNN	Fast R-CNN	Faster R-CNN
Test time per image (with proposals)	50 seconds	2 seconds	**0.2 seconds**
(Speedup)	1x	25x	**250x**
mAP (VOC 2007)	66.0	**66.9**	**66.9**

86